



# *TrailsCommunity*

Rapport de projet - Assitant Offroad

BELLANGER Stéphane

MONNIER Ysée

DESHORS Yann

Novembre 2016 - Decembre 2016

# Table des matières

<b>I</b>	<b>Spécification des exigences logicielles</b>	<b>4</b>
<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Objectifs . . . . .	5
1.2	Portée . . . . .	5
1.3	Définitions . . . . .	5
1.4	Références . . . . .	6
1.5	Aperçu . . . . .	6
<b>2</b>	<b>Description générale</b>	<b>7</b>
2.1	Perspective du produit . . . . .	7
2.2	Fonctions du produit . . . . .	7
2.3	Caractéristique de l'utilisateur . . . . .	8
2.4	Contraintes . . . . .	8
2.5	Hypothèses et dépendances . . . . .	9
2.6	Répartition des besoins . . . . .	9
<b>3</b>	<b>Exigences particulières</b>	<b>10</b>
3.1	Exigences externe de l'interface . . . . .	10
3.2	Les interfaces utilisateurs . . . . .	10
3.3	Interfaces matérielles . . . . .	13
3.4	Interfaces logicielles . . . . .	13
3.5	Interfaces de communication . . . . .	14
<b>4</b>	<b>Exigences fonctionnelles</b>	<b>15</b>
4.1	Visiteur . . . . .	15
4.1.1	Exigence fonctionnelle 1.1 . . . . .	15
4.1.2	Exigence fonctionnelle 1.2 . . . . .	15
4.1.3	Exigence fonctionnelle 1.3 . . . . .	15
4.1.4	Exigence fonctionnelle 1.4 . . . . .	16
4.2	Utilisateur . . . . .	16
4.2.1	Exigence fonctionnelle 1.5 . . . . .	16
4.2.2	Exigence fonctionnelle 1.6 . . . . .	16
4.3	Organisateur . . . . .	17
4.3.1	Exigence fonctionnelle 1.8 . . . . .	17

<b>II</b>	<b>Serveur TrailsCommunity</b>	<b>18</b>
<b>1</b>	<b>Introduction</b>	<b>19</b>
1.1	Conventions . . . . .	19
1.2	Codes statut . . . . .	19
<b>2</b>	<b>Ressources</b>	<b>20</b>
2.1	Authentification . . . . .	20
2.1.1	Requête . . . . .	20
2.1.2	Réponse . . . . .	20
2.2	Inscription . . . . .	21
2.2.1	Requête . . . . .	21
2.2.2	Réponse . . . . .	22
2.3	Ajout d'une session . . . . .	22
2.3.1	Requête . . . . .	22
2.3.2	Réponse . . . . .	23
2.4	Liste des sessions . . . . .	23
2.4.1	Requête . . . . .	23
2.4.2	Réponse . . . . .	24
2.5	Session . . . . .	25
2.5.1	Requête . . . . .	25
2.5.2	Réponse . . . . .	26
2.6	Mis à jour d'une session . . . . .	27
2.6.1	Requête . . . . .	27
2.6.2	Réponse . . . . .	27
2.7	Rejoindre une session . . . . .	28
2.7.1	Requête . . . . .	28
2.7.2	Réponse . . . . .	28
2.8	Ajout d'un waypoint . . . . .	28
2.8.1	Requête . . . . .	28
2.8.2	Réponse . . . . .	29

# Table des figures

3.1	Maquette de l'IHM pour la connexion . . . . .	10
3.2	Maquette de l'IHM pour la création d'un compte utilisateur . . . . .	11
3.3	Maquette de l'IHM pour l'affichage des différentes sessions . . . . .	11
3.4	Maquette de l'IHM pour la modification des données utilisateurs . . . . .	12
3.5	Maquette de l'IHM pour la création d'une session . . . . .	12
3.6	Maquette de l'IHM pour l'affichage des différentes positions des participants et de réponse au partage d'un waypoint. . . . .	12
3.7	Maquette de l'IHM pour le chat d'une session courante . . . . .	13
3.8	Maquette de l'IHM pour l'affichage des statistiques d'une session courante . . . . .	13

# Première partie

## Spécification des exigences logicielles

# Chapitre 1

## Introduction

Cette section fournit une vue d'ensemble de tout ce qui est inclus dans ce document SRS. De plus, l'objectif de ce document est décrit et une liste de définitions est fournie.

### 1.1 Objectifs

L'objectif de ce document est de fournir une description détaillée des conditions requises pour l'application TrailsCommunity. Il illustrera le but et la description complète du développement du système. Il expliquera également les contraintes du système, l'interface et les interactions avec les applications externes. Ce document est principalement destiné à être proposé à un client pour son approbation. C'est aussi une référence pour le développement de la première version du système pour l'équipe de développement.

### 1.2 Portée

TrailsCommunity est une application mobile GPS qui permet aux utilisateurs de connaître en temps réel le positionnement des autres participants d'une activité de pleine air en cours. L'ensemble des activités sont regroupées en type qui seront limitées à une liste pré-définie. L'application doit être gratuite et simple d'utilisation. L'organisateur de l'activité devra fournir deux adresses. Une pour le point de départ de l'application et la deuxième pour le point d'arrivée. La création des sessions se fera directement sur l'application mobile. En outre, le logiciel a besoin à la fois d'Internet et de connexion GPS pour récupérer et afficher les résultats. Tous les systèmes d'informations sont conservés dans une base de données, qui se trouve sur un serveur web. Le logiciel interagit également avec l'API Google Map qui permettra d'afficher et de tracer les différents parcours des utilisateurs. De plus, le téléphone doit aussi comprendre un GPS pour pouvoir localiser l'utilisateur à tout moment. L'application a également la capacité de représenter à la fois des informations sommaires et détaillées des différentes statistiques de l'utilisateur courant mais aussi des autres participants de la session en cours.

### 1.3 Définitions

session    une session est une activité disponible dans l'application

visiteur personne n'étant pas identifier

utilisateur personne identifié à l'application. Il possède les choix de créer ou de rejoindre une session

organisateur utilisateur ayant créé une session et pouvant la gérer

participant utilisateur qui a rejoint une session

waypoint désigne un point de la route a atteindre ou doit avoir lieu un changement de cap

## 1.4 Références

The Institute of Electrical and Electronic Engineer NY USA IEEE Recommended Practice for SRS,  
<http://www.utdallas.edu/~chung/RE/IEEE830-1993.pdf>

Chalmers IEEE standards - SRS Example, 2010, [http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs\\_example\\_2010\\_group2.pdf](http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf)

Droid5 Informatics Pvt Ltd androidhive, 2016, <http://www.androidhive.info>

Oracle Java 8 Documentation, 2016, <https://docs.oracle.com/javase/8/docs/api/>

Google Android Documentation, 2016, <https://developer.android.com/guide>

James Britt Ruby Doc, 2016, <http://ruby-doc.org>

Rails Community Ruby On Rails Documentation, 2016, <http://rubyonrails.org>

## 1.5 Aperçu

Le reste de ce document comprend quatres chapitres. Le second fournit une vue d'ensemble de la fonctionnalité du système et de l'interaction du système avec d'autres systèmes. En outre, le chapitre mentionne également les contraintes du systèmes et les hypothèses sur e produit. Le troisième chapitre fournit la conception et la description du serveur.

Différentes techniques de spécification sont utilisées afin de préciser les exigences plus précisément pour différents publics. Cependant, le lecteur principal sera le correcteur du projet. Le quatrième chapitre regroupe l'ensemble des annexes du projet. Ce sont en globalité des diagrammes UML. Ils permettent soit une meilleurs finesse de compréhension du système soit, une meilleurs spécification métier du serveur.

# Chapitre 2

## Description générale

### 2.1 Perspective du produit

Ce système se compose de deux parties : une application mobile et un serveur web. L'application Android sera utilisée pour la gestion des sessions. Elle permettra d'afficher en fonction d'une session sélectionner, sur une carte Google map les différentes positions des participants.

Le serveur sera tant qu'à lui capable de gérer la récolte et la gestion des données envoyées par l'application.

L'application mobile devra communiquer avec un GPS qui se situe au sein du mobile. Le GPS fournira les emplacements de l'utilisateur et des autres participants. Mais aussi des statistiques sur l'utilisateur courant ainsi que chat pour permettre aux différents membres de pouvoir communiquer ensemble.

De manière transparente comme il s'agit d'un produit axé sur la récolte de données, il lui faudra une base de données. La communication avec la base de données se passera via internet.

### 2.2 Fonctions du produit

Avec l'application mobile, l'utilisateur pourra créer un compte utilisateur. Cette première étape va permettre de collecter différentes informations primordiales sur l'utilisateur.

Après la réalisation de cette première étape, les utilisateurs pourront sélectionner des sessions. Le résultat sera basé sur les sessions que l'utilisateur sélectionne. Il existe 2 types de sessions. Les sessions actives, elles sont encore réalisables et ne sont pas encore clôturées. Cela veut donc dire que leurs dates de fin ne sont pas dépassées. Dans ces sessions, un chat est disponible en temps réel est disponible permettant de communiquer avec tous les participants.

Les sessions clôturées dont l'utilisateur a participé. Elles sont classées dans la section historique qui permet à l'utilisateur courant de pouvoir revisualiser ses anciennes statistiques.



L'application va permet après sélection d'une session, d'afficher une carte Google map ainsi que ensemble de point permettant de décrire le déplacement des différents utilisateurs actif de la session. De plus, les sessions pourront être directement créée depuis l'application.

## 2.3 Caractéristique de l'utilisateur

Il existe quatre types d'utilisateurs qui interagissent avec le système : le visiteur, l'utilisateur, le participant et l'organisateur. Chacun de ces trois types d'utilisateurs a une utilisation différente du système afin que chacun ait leurs propres exigences.

Les visiteurs de l'application mobile ne peuvent utiliser TrailsCommunity si il ne sont pas connecter. Cela signifie que le visiteur doit être en mesure de pouvoir créer un compte utilisateur et ainsi pouvoir se connecter.

L'utilisateur qui sont des visiteurs connecter peuvent alors visualiser l'ensemble des sessions actives mais aussi leur historique de session. De plus, ils peuvent modifier leurs coordonnées personnelles.

Les participants sont des utilisateurs qui ont rejoint une session active, ils peuvent alors ajouter un waypoint et le partager à l'ensemble des participants. Ils peuvent aussi communiquer via un chat en temps réel. De plus, ils peuvent consulter leurs statistiques en cours.

Enfin, les organisateurs sont des utilisateurs qui ont créée une nouvelle session. En effet, ce sont eux dont l'impulsion est venu pour organiser une activités. Ce sont eux qui vont pouvoir faire partager le mot de passe de la session pour pouvoir la rejoindre.

## 2.4 Contraintes

L'application mobile est limitée au système de navigation GPS du téléphone portable. Comme il existe plusieurs système de fabricants de GPS, la précision n'est pas la même pour chacun d'entre eux. En outre, il peut y avoir une différences de navigation en fonction des téléphone.

La connexion internet est également une contrainte pour l'application. Puisqu'elle récupère les données de la base de données, il est crucial qu'il existe une connexion. De plus en fonction du réseau, il est possible d'avoir une différence du temps de réception des données plus lente. Il existera une fonctionnalité offline qui permettra entre autre de pouvoir basculer sur un nouveau protocole pour que les utilisateurs puissent continuer à communiquer entre eux. L'ensemble des données récoltés sont alors stockées dans une base de données internet du mobile. Lors de la récupération du réseau internet, l'application va alors recevoir l'ensemble des données collectés en dur et va reprendre son état d'envoi naturel.

## 2.5 Hypothèses et dépendances

Le produit sera toujours utilisé sur les téléphones mobiles Android qui ont assez de performances. Si le téléphone ne dispose pas de ressources matérielles suffisantes pour l'application. Elle peut ne pas fonctionner comme prévu ou même pas du tout.

Une autre hypothèse est que les composants GPS de tous les téléphones fonctionnent de la même manière. Si le téléphone a un système GPS qui diffère des normes, l'application doit être spécifiquement adaptée à chaque interface.

## 2.6 Répartition des besoins

Dans le cas où le projet est retardé, certaines exigences pourraient être transférées à la prochaine version de l'application. Les exigences avec une priorité haute doivent être développées lors du rendu de la première version de l'application.

# Chapitre 3

## Exigences particulières

Cette section contient toutes les exigences fonctionnelles et de qualité du système. Il donne une description du système et de toutes ses caractéristiques.

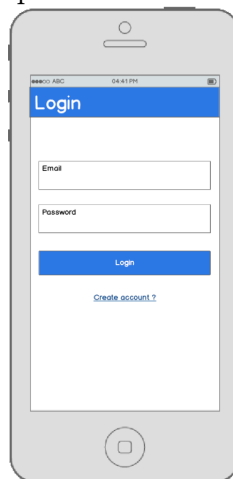
### 3.1 Exigences externe de l'interface

Cette section fournit une description détaillée de toutes les entrées et sorties du système. Il donne également une description des interfaces homme machine et fournit des prototypes de base de l'interface pour l'utilisateur.

### 3.2 Les interfaces utilisateurs

Un visiteur de l'application mobile devrait voir la page d'ouverture de session quand il ouvre l'application. Pour la connexion au compte utilisateur, il lui est absolument nécessaire de posséder un compte utilisateur composé d'une adresse mail et du mot de passe associé.

FIGURE 3.1 – Maquette de l'IHM pour la connexion



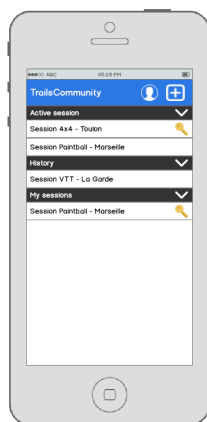
Pour la création d'un compte utilisateur, il est nécessaire de remplir l'ensemble des champs de l'interface. Toutefois, il sera possible de modifier son pseudonyme ainsi que son numéro de téléphone.

FIGURE 3.2 – Maquette de l’IHM pour la création d’un compte utilisateur

Maquette d'un écran de smartphone pour la création d'un compte utilisateur. L'écran est intitulé "Register" en haut à gauche. Il contient cinq champs de saisie : "Email", "Nickname", "Password", "Password again", et un champ pour le numéro de téléphone avec un préfixe "+33" et l'étiquette "telephone number". Un bouton "Register" est situé en bas.

Après la connexion à un compte utilisateur, celui-ci peut alors choisir entre des sessions actives ou des sessions clôturer qui sont dans la liste des historiques. De plus, le bouton avec l’icône d’une personne permet d’accéder aux données utilisateurs. L’icône plus permet de créer une nouvelle session.

FIGURE 3.3 – Maquette de l’IHM pour l’affichage des différentes sessions

Maquette d'un écran de smartphone pour l'affichage des sessions. L'écran est intitulé "TriaCommunity" en haut à gauche. Il contient une liste de sessions avec des icônes de personne et de plus. Les sessions sont : "Active session", "Session 4x4 - Toulon", "Session Paritbal - Marseille", "History", "Session VTT - La Garde", "My sessions", et "Session Paritbal - Marseille".

Pour la modification des données utilisateurs, seulement le pseudonyme et le numéro de téléphone peuvent être modifié.

La création d’une session possède de nombreux champs texte obligatoire. Le champs du mot de passe ne sera connu que par l’organisateur de la session.

FIGURE 3.4 – Maquette de l’IHM pour la modification des données utilisateurs

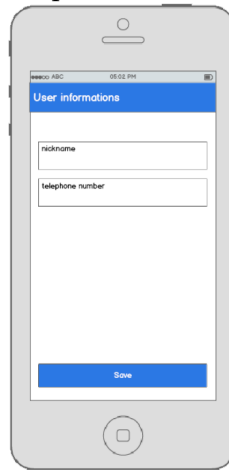
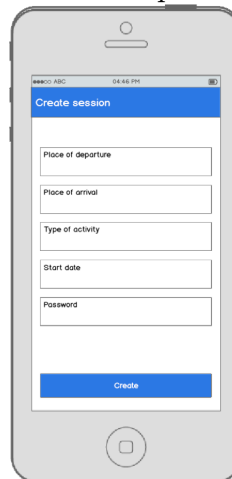


FIGURE 3.5 – Maquette de l’IHM pour la création d’une session



La maquette principal est l’affichage des différentes positions des participants de la session courante. Sur cette maquette nous pouvons aussi voir comment l’utilisateur peut accepter ou non un waypoint partagé par un autre participant.

La maquette suivante présente le chat d’une session courante.

Cette figure montre l’affichage des statistiques du session courante.

### 3.3 Interfaces matérielles

Puisque l’application mobile ne comportent pas de matériel désigné, il n’y a pas d’interfaces matérielles directes. Le GPS physique est géré directement par l’application et la connexion matérielle au serveur est géré par le système d’exploitation sous-jacent sur le téléphone mobile.

FIGURE 3.6 – Maquette de l'IHM pour l'affichage des différentes positions des participants et de réponse au partage d'un waypoint.

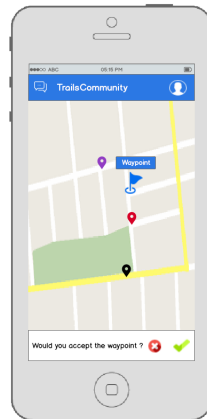
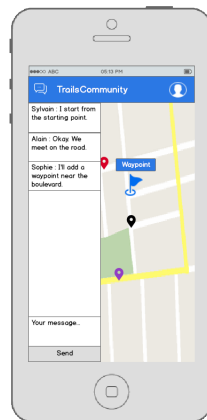


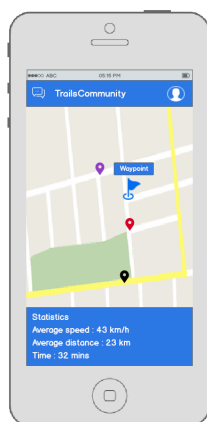
FIGURE 3.7 – Maquette de l'IHM pour le chat d'une session courante



### 3.4 Interfaces logicielles

L'application mobile communique avec l'application GPS pour obtenir les informations géographiques. Sur la localisation de l'utilisateur et la représentation visuelle de celle-ci, l'utilisation du serveur est primordiale afin d'obtenir en temps réel les informations sur les participants. La communication entre la base de données et l'application se compose des opérations d'écriture et de lecture.

FIGURE 3.8 – Maquette de l'IHM pour l'affichage des statistiques d'une session courante



## 3.5 Interfaces de communication

La communication entre les différentes parties du système est importante car elles dépendent les unes des autres. Cependant, la manière dont la communication est réalisée n'est pas importante pour le système et est donc traitée par les systèmes d'exploitation pour l'application mobile.

# Chapitre 4

## Exigences fonctionnelles

Cette section comprend les exigences qui spécifient toutes les actions fondamentales du système logiciel.

### 4.1 Visiteur

#### 4.1.1 Exigence fonctionnelle 1.1

**ID : FR1**

**TITRE :** Télécharger l'application mobile

**DESCRIPTION :** Un utilisateur doit être en mesure de télécharger l'application mobile via le Google play store. L'application doit être libre de téléchargement.

**BUT :** Pour qu'un utilisateur puisse télécharger l'application mobile.

**DÉPENDANCE :** Aucune.

#### 4.1.2 Exigence fonctionnelle 1.2

**ID : FR2**

**TITRE :** Télécharger et notifier les utilisateurs des nouvelles versions.

**DESCRIPTION :** Lorsqu'une nouvelle version/mise à jour est publiée, l'utilisateur doit les vérifier manuellement. Le téléchargement de la nouvelle version devrait se faire via le téléphone mobile de la même manière que le téléchargement de l'application mobile.

**BUT :** Pour que l'utilisateur puisse télécharger une nouvelle version.

**DÉPENDANCE :** FR1.

#### 4.1.3 Exigence fonctionnelle 1.3

**ID : FR3**

**TITRE :** Enregistrement du visiteur sur l'application mobile.

**DESCRIPTION :** Étant donné qu'un utilisateur a téléchargé l'application mobile, l'utilisateur devrait pouvoir s'enregistrer via l'application mobile. L'utilisateur doit fournir en premier lieu son surnom puis vient en deuxième, adresse mail ainsi que son mot de passe et pour finir son numéro de téléphone.



BUT : Pour qu'un visiteur puisse s'inscrire sur l'application mobile.  
DÉPENDANCE : FR1.

#### **4.1.4 Exigence fonctionnelle 1.4**

**ID : FR4**

TITRE : Connexion du visiteur sur l'application mobile.

DESCRIPTION : Étant donné qu'un visiteur s'est enregistré, il doit pouvoir se connecter à l'application mobile via son adresse mail et son mot de passe. Les informations d'ouverture de session seront stockées dans la base de données.

BUT : Pour qu'un visiteur puisse se connecter sur l'application mobile.

DÉPENDANCE : FR1, FR3.

### **4.2 Utilisateur**

#### **4.2.1 Exigence fonctionnelle 1.5**

**ID : FR5**

TITRE : Un utilisateur peut rejoindre une session.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit à le choix entre 3 types de sessions. Le premier type est les sessions actives, elles sont soit privée ou publique. Si elles sont privées, l'utilisateur doit connaître le mot de passe pour les rejoindre. Ensuite, le deuxième type est les sessions dont il a déjà participé, cela fonctionne comme un historique. La dernière est les sessions que l'utilisateurs à créées.

BUT : Pour qu'un organisateur puisse rejoindre une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

#### **4.2.2 Exigence fonctionnelle 1.6**

**ID : FR6**

TITRE : Un utilisateur peut créer une session.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit pouvoir créer une nouvelle session dans l'application. Les informations de la nouvelle session créée seront stockées dans la base de données du serveur distant. L'utilisateur doit fournir le lieu de départ ainsi que le lieu d'arrivée. Le format d'une adresse lui sera demandé. L'application convertira cette adresse en coordonnées GPS. De plus, il devra indiquer le type d'activité ainsi que la date de départ. Pour finir, l'organisateur aura la possibilité de verrouiller la session par un mot de passe qu'il divulguera à l'ensemble des participants.

BUT : Pour qu'un organisateur puisse créer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

#### **4.2.3 Exigence fonctionnelle 1.5**

**ID : FR5**

TITRE : Un utilisateur peut se déconnecter de l'application.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit avoir le choix, à tout moment de pouvoir se déconnecter de l'application. Il sera alors redirigé vers la page de connexion.

BUT : Pour qu'un organisateur puisse se déconnecter de l'application mobile.

DÉPENDANCE : FR1, FR3.

## 4.3 Participant

### 4.3.1 Exigence fonctionnelle 1.5

**ID : FR5**

TITRE : Un participant peut ajouter un waypoint dans une session active.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut ajouter un waypoint. Lorsque l'utilisateur maintient longuement son doigt sur l'écran de son téléphone, un marker est alors ajouté. Le waypoint est alors diffusé à tout les autres participants.

BUT : Pour qu'un participant puisse ajouter un waypoint dans une session active.

DÉPENDANCE : FR1, FR3.

## 4.4 Organisateur

### 4.4.1 Exigence fonctionnelle 1.8

**ID : FR8**

TITRE : Un organisateur peut gérer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté, il doit pouvoir gérer ses sessions. Touts les champs remplit précédemment lors de la création de la session sont modifiables. Les informations de la session seront stockées dans la base de données.

BUT : Pour qu'un organisateur puisse gérer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

### 4.4.2 Exigence fonctionnelle 1.9

**ID : FR9**

TITRE : Un organisateur peut clôturer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connectée à sélectionné une de ses session créée précédemment. Il doit pouvoir clôturer la session sélectionné. Celle-ci n'est alors plus présente dans les sessions actives de l'application sauf pour les participants ou la session est alors disponible dans leurs historique.

BUT : Pour qu'un organisateur puisse clôturer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

## 4.5 Application

### 4.5.1 Exigence fonctionnelle 1.5

**ID : FR5**

TITRE : L'application doit pouvoir enregistrer un parcours effectué sans connexion internet.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne. Une base de données interne a l'application va alors stockées l'ensemble des données.

BUT : Pour que l'application puisse pouvoir enregistrer un parcours effectué sans connexion internet.

DÉPENDANCE : FR1, FR3, FR4.

#### **4.5.2 Exigence fonctionnelle 1.5**

##### **ID : FR5**

TITRE : Le chat de l'application doit pouvoir fonctionné en mode hors ligne.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne du chat. L'application va alors basculé sur un le protocole SMS pour que les participants puissent continués de communiquer ensemble.

BUT : Pour que le chat de l'application puisse fonctionner en mode hors ligne.

DÉPENDANCE : FR1, FR3, FR4.

#### **4.5.3 Exigence fonctionnelle 1.5**

##### **ID : FR5**

TITRE : L'application doit pouvoir fonctionné en mode hors ligne.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne du chat. L'application va alors basculé sur un le protocole SMS pour que les participants puissent continués de communiquer ensemble.

BUT : Pour que le chat de l'application puisse fonctionner en mode hors ligne.

DÉPENDANCE : FR1, FR3, FR4.

# Deuxième partie

## Serveur TrailsCommunity

# Chapitre 1

## Introduction

### 1.1 Conventions

- **Client** : Application client.
- **Statut** : Code réponse de la requête HTTP
- Toutes les réponses possibles sont listés sous la catégorie 'Réponse'
- Toutes les réponses sont du format JSON

### 1.2 Codes statut

Tous les codes de statut sont des standards HTTP. Cette API utilisera les suivants :

- *2XX* - Succès de la requête
- *4XX* - Erreur du côté client
- *5XX* - Erreur du côté serveur

Code statut	Description
200	Requête traitée avec succès
201	Requête traitée avec succès et création d'un document
202	Requête traitée avec succès mais pas d'information à renvoyer
400	La syntaxe de la requête est erronée
401	Une authentification est nécessaire pour accéder à la ressource
404	Ressource non trouvée
500	Erreur interne du serveur
503	Service temporairement indisponible ou en maintenance

# Chapitre 2

## Ressources

### 2.1 Authentification

#### 2.1.1 Requête

Méthode	URL
<i>POST</i>	api/login/auth_token

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>JSON</i>	<pre>{   "auth": {     "email": "mail@mail.com",     "password": "FZ,f235nDE",   } }</pre>	<i>String</i> <i>String</i>

*email* : correspond à l'adresse mail valide de l'utilisateur

*password* : mot de passe de l'utilisateur

#### 2.1.2 Réponse

Statut	Réponse
200	{ "jwt": <Authorization> }
401	
500	

*Authorization(string)* - Tous les autres appels nécessitant une authentification doivent avoir cette clé dans le header.

## 2.2 Inscription

### 2.2.1 Requête

Enregistrement d'un utilisateur dans le système.

Méthode	URL
<i>POST</i>	api/users/

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>JSON</i>	{ "nickname": "Arthur83", "email": "apaul@custom.com", "phone_number": "+33651678908", "password": "myPassword33", "password_confirmation": "myPassword33" }	<i>String</i> <i>String</i> <i>String</i> <i>String</i> <i>String</i>

*nickname* : pseudonyme de l'utilisateur

*email* : email de l'utilisateur non nul et valide

*password* : mot de passe d'utilisateur

*password\_confirmation* : confirmation du mot de passe de l'utilisateur. Il doit être identique à *password*.

## 2.2.2 Réponse

Statut	Réponse
201	<pre>{   "data": {     "id": 2,     "nickname": "Arthur83",     "phone_number": "+33651678908",     "current_session_id": null   } }</pre>
400	<pre>{   "errors": {     "nickname": [       "has already been taken"     ],     "email": [       "is invalid"     ]   } }</pre>
500	

## 2.3 Ajout d'une session

### 2.3.1 Requête

Permet d'ajouter une nouvelle session dans le serveur. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>POST</i>	<i>api/sessions/</i>

Type	Paramètres	Valeurs
<i>HEADER</i>	<i>Application/json</i>	<i>String</i>
<i>HEADER</i>	<i>Authorization</i>	<i>String</i>
<i>JSON</i>	<pre>{   "activity": 1,   "password": "lockSessionPsw",   "departure_place": "43.179363;5.717782",   "arrival_place": "43.191168;5.730819",   "start_date": "2016-11-20" }</pre>	<i>Integer</i> <i>String</i> <i>String</i> <i>String</i> <i>Date</i>



*activity* : identifiant de l'activité. (1 : randonnée, 2 : vélo, ...)  
*password* : mot de passe de la session, permet de restreindre l'entrée.  
*departure\_place* : lieu de départ. Couple de position géographique (*latitude*; *longitude*)  
*arrival\_place* : lieu d'arrivée. Couple de position géographique (*latitude*; *longitude*)

## 2.3.2 Réponse

Statut	Réponse
200	<pre>{   "data": {     "id": 5,     "activity": 1,     "departure_place": "43.179363;5.717782",     "arrival_place": "43.191168;5.730819",     "start_date": "2016-11-20",     "close": false,     "lock": true,     "user": {       "id": 1,       "nickname": "Raptor234",       "phone_number": "+33623569450",       "current_session_id": null     },     "coordinates": [],     "waypoints": []   } }</pre>
400	<pre>{   "errors": {     "activity": [       "can't be blank",       "is not a number"     ]   } }</pre>
401	
500	

## 2.4 Liste des sessions

### 2.4.1 Requête

Permet de récupérer la liste des sessions du système. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>GET</i>	api/sessions/

Type	Paramètres	Valeurs
<i>HEADER</i>	Authorization	<i>String</i>

## 2.4.2 Réponse

Statut	Réponse
200	<pre>{   "data": [     {       "id": 1,       "activity": 1,       "departure_place": "43.179363;5.717782",       "arrival_place": "43.191168;5.730819",       "start_date": "2016-11-20",       "close": false,       "lock": true,       "user": {         "id": 2,         "nickname": "Arthur83",         "phone_number": "+33651678908",         "current_session_id": null       }     },     {       "id": 2,       "activity": 3,       "departure_place": "43.179363;5.717782",       "arrival_place": "43.191168;5.730819",       "start_date": "2016-11-20",       "close": false,       "lock": true,       "user": {         "id": 2,         "nickname": "Arthur83",         "phone_number": "+33651678908",         "current_session_id": null       }     }   ] }</pre>
401	
500	

## 2.5 Session

### 2.5.1 Requête

Permet de récupérer une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>GET</i>	<code>api/sessions/:id</code>

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	<code>id</code>	<i>Integer</i>
<i>HEADER</i>	<code>Authorization</code>	<i>String</i>

*id* : identifiant de la session voulu

## 2.5.2 Réponse

Statut	Réponse
200	<pre>{   "data": {     "id": 2,     "activity": 3,     "departure_place": "43.179363;5.717782",     "arrival_place": "43.191168;5.730819",     "start_date": "2016-11-20",     "close": false,     "lock": true,     "user": {       "id": 2,       "nickname": "Arthur83",       "phone_number": "+33651678908",       "current_session_id": null     },     "coordinates": [       {         "latitude": 43.179363,         "longitude": 5.717782,         "user_id": 1       }     ],     "waypoints": [       {         "latitude": 43.179363,         "longitude": 5.717782       }     ]   } }</pre>
401	
404	<pre>{   "errors": {     "session": "Record Not Found"   } }</pre>
500	

## 2.6 Mis à jour d'une session

### 2.6.1 Requête

Permet de mettre à jour session spécifique selon un identifiant (ID) dans le système. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>PUT</i>	<code>api/sessions/:id</code>

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	<pre>{   "close": true }</pre>	<i>boolean</i>

*close* : boolean déterminant si la session doit être fermée.

### 2.6.2 Réponse

Statut	Réponse
200	<pre>{   "data": {     "id": 7,     "activity": 3,     "departure_place": "43.179363;5.717782",     "arrival_place": "43.191168;5.730819",     "start_date": "2016-11-20",     "close": true,     "lock": true,     "user": {       "id": 2,       "nickname": "Arthur83",       "phone_number": "+33651678908",       "current_session_id": 1     },     "coordinates": [],     "waypoints": []   } }</pre>
401	
500	

## 2.7 Rejoindre une session

### 2.7.1 Requête

Permet de à l'utilisateur de rejoindre une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>GET</i>	<code>api/sessions/:id/join</code>

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	<code>id</code>	<i>Integer</i>
<i>HEADER</i>	<code>Authorization</code>	<i>String</i>

*id* : identifiant de la session voulu

### 2.7.2 Réponse

Statut	Réponse
200	
401	
404	<pre>{   "errors": {     "session": "Record Not Found"   } }</pre>
500	

## 2.8 Ajout d'un waypoint

### 2.8.1 Requête

Permet de à l'utilisateur d'ajouter une nouveau waypoint pour une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth\_token*

Méthode	URL
<i>POST</i>	<code>api/sessions/:id/waypoint</code>

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	id	<i>Integer</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	<pre>{     "latitude": 43.179363,     "longitude": 5.717782 }</pre>	<i>Double</i> <i>Double</i>

*id* : identifiant de la session voulu

## 2.8.2 Réponse

Statut	Réponse
200	
401	
404	<pre>{   "errors": {     "session": "Record Not Found"   } }</pre>
500	